

# Overture Demo Introduction

Kyle Chand and Bill Henshaw

Lawrence Livermore National Laboratory,  
Livermore, CA, USA  94551

[www.llnl.gov/casc/Overture](http://www.llnl.gov/casc/Overture)

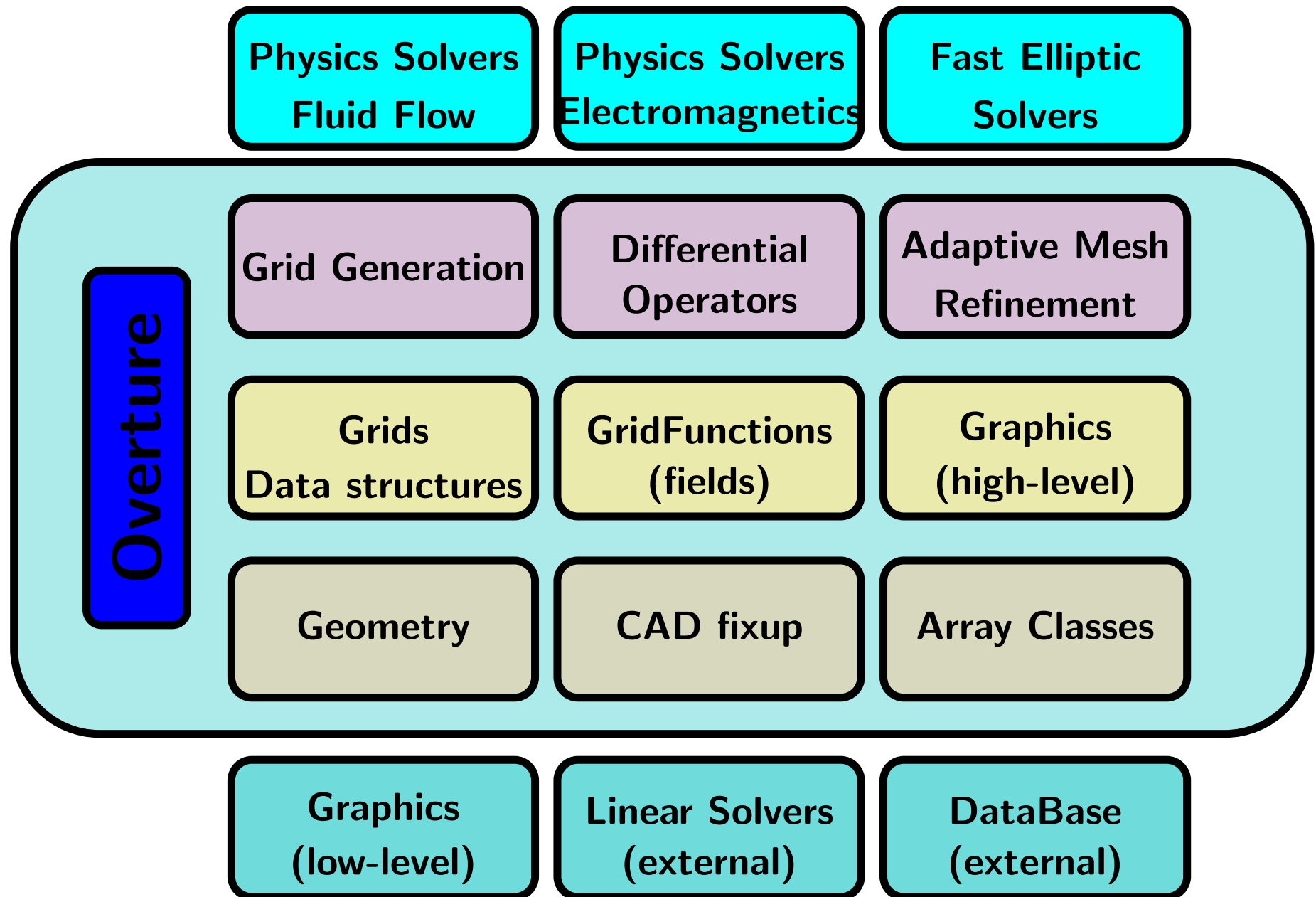
9th Overset Grid Symposium, Penn State University, Pennsylvania  October  
2007. LLNL-PRES-407076.

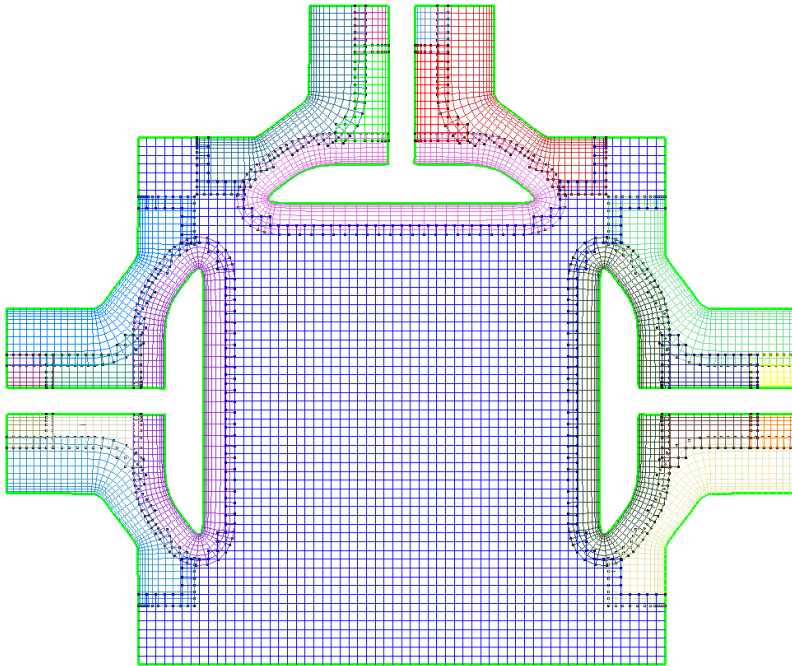
# Overture is toolkit for solving partial differential equations on structured, overlapping and hybrid grids.

## Key features:

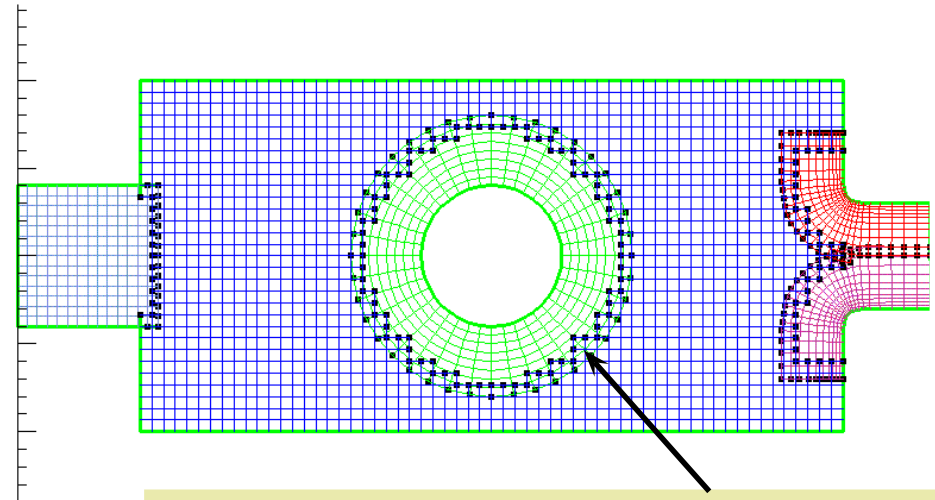
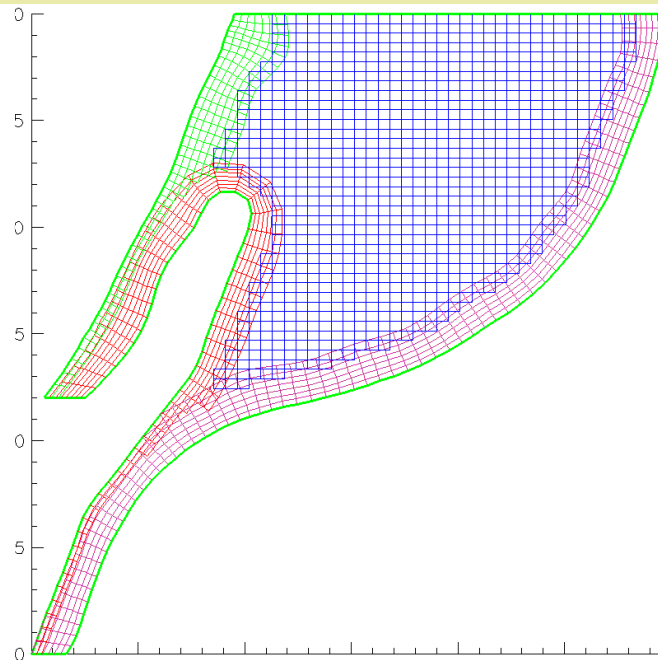
- provides a high level C++ interface for rapid prototyping of PDE solvers.
- built upon optimized C and fortran kernels.
- provides a library of finite-difference operators: conservative and non-conservative, 2nd, 4th, 6th and 8th order accurate approximations.
- support for moving grids.
- support for block structured adaptive mesh refinement (AMR).
- extensive grid generation capabilities.
- CAD fixup tools (for CAD from IGES files).
- interactive graphics and data base support (HDF).
- PDE solvers built upon Overture include:
  - cginas: incompressible Navier-Stokes with heat transfer.
  - cgcns: compressible Navier-Stokes, reactive Euler equations.
  - cgmp: multi-physics solver.
  - cgmx: time domain Maxwell's equations solver.

# The Overture Framework supports Physics Codes





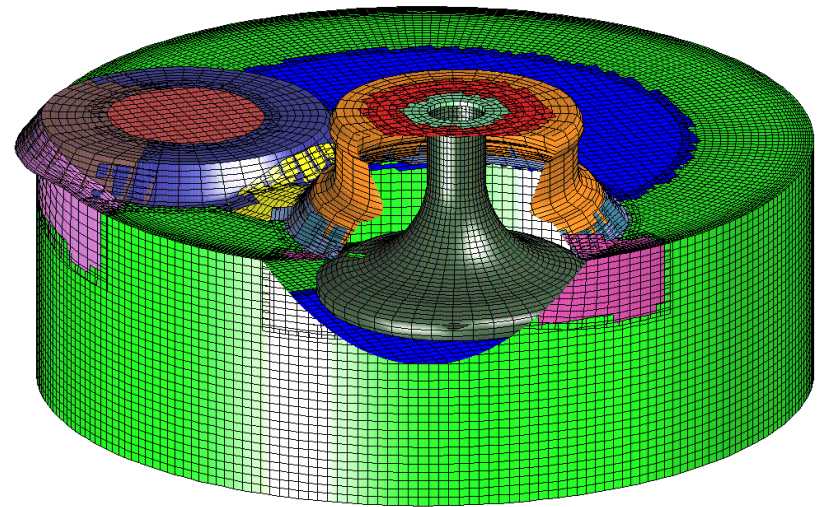
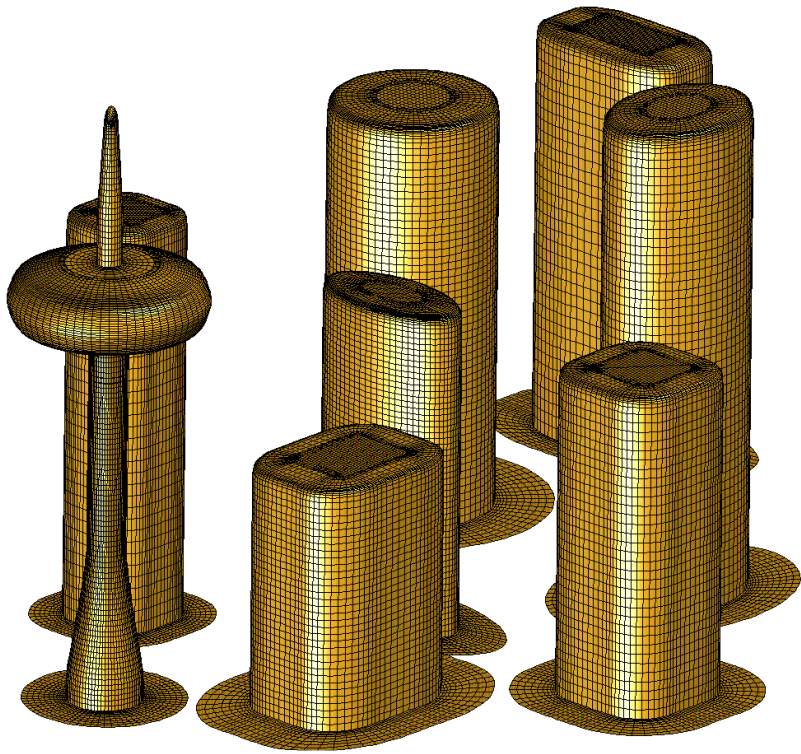
Sample 2D overlapping grids



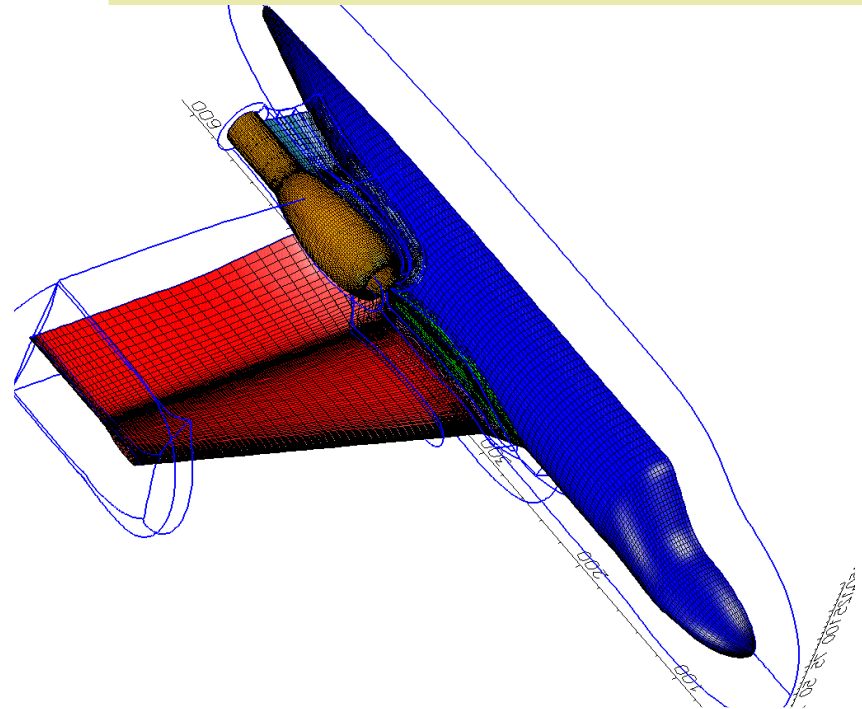
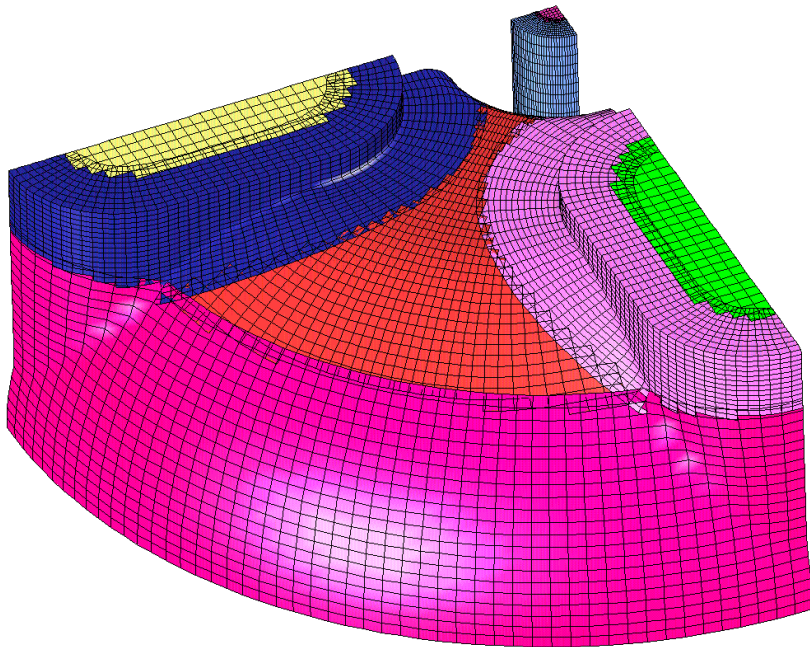
Solutions coupled by interpolation



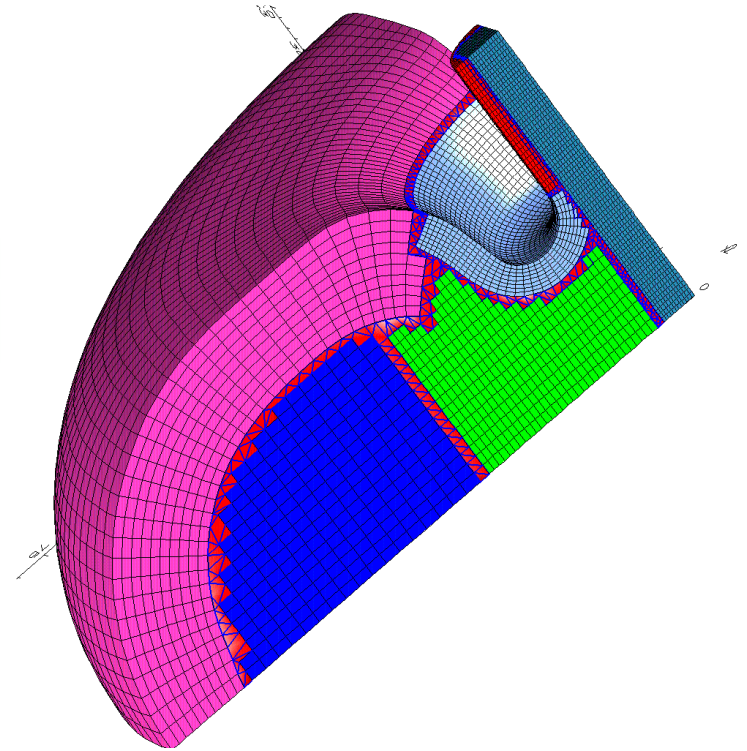
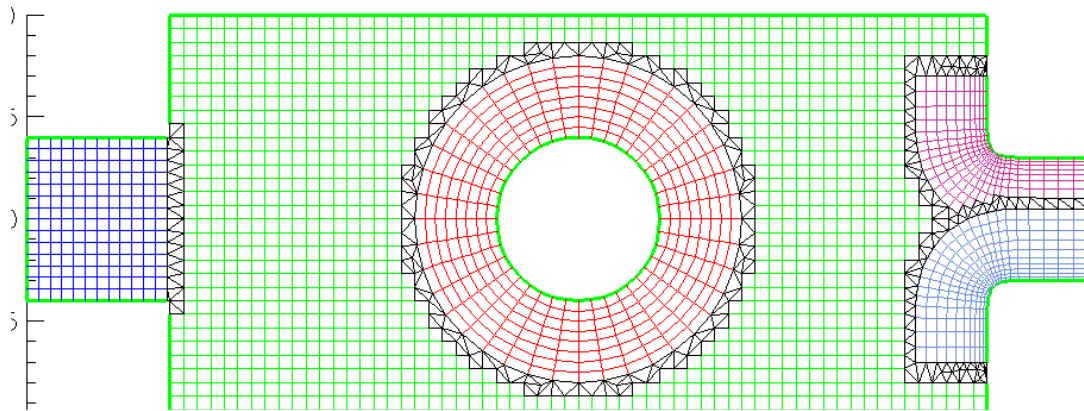




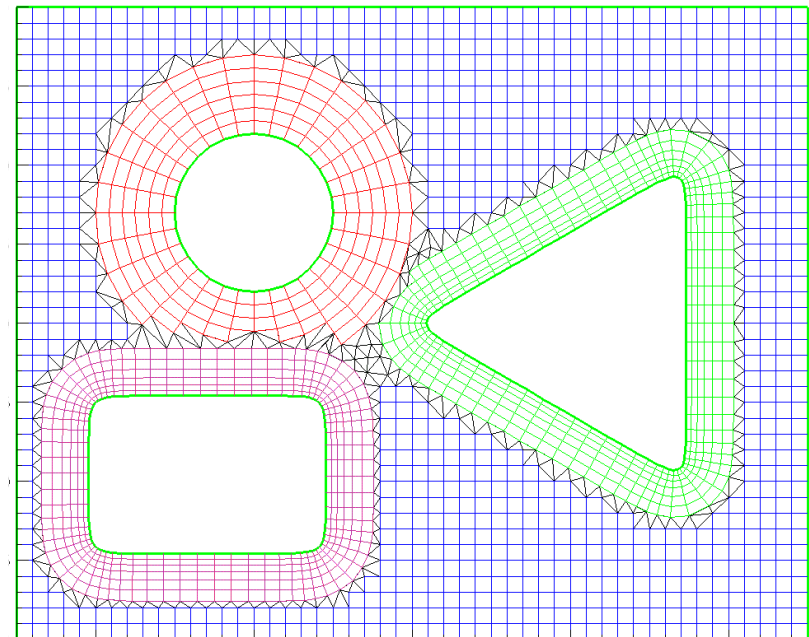
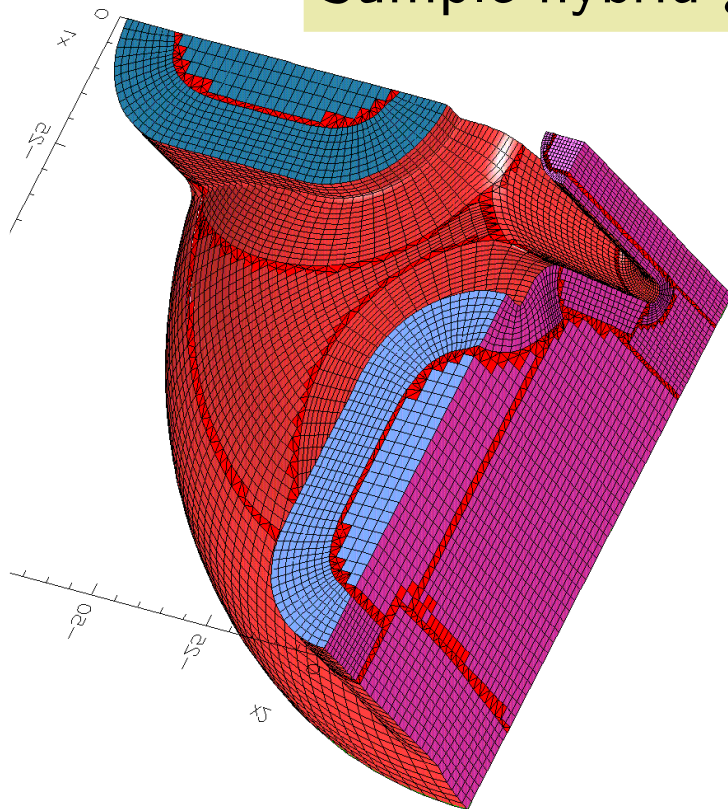
Sample 3D overlapping grids



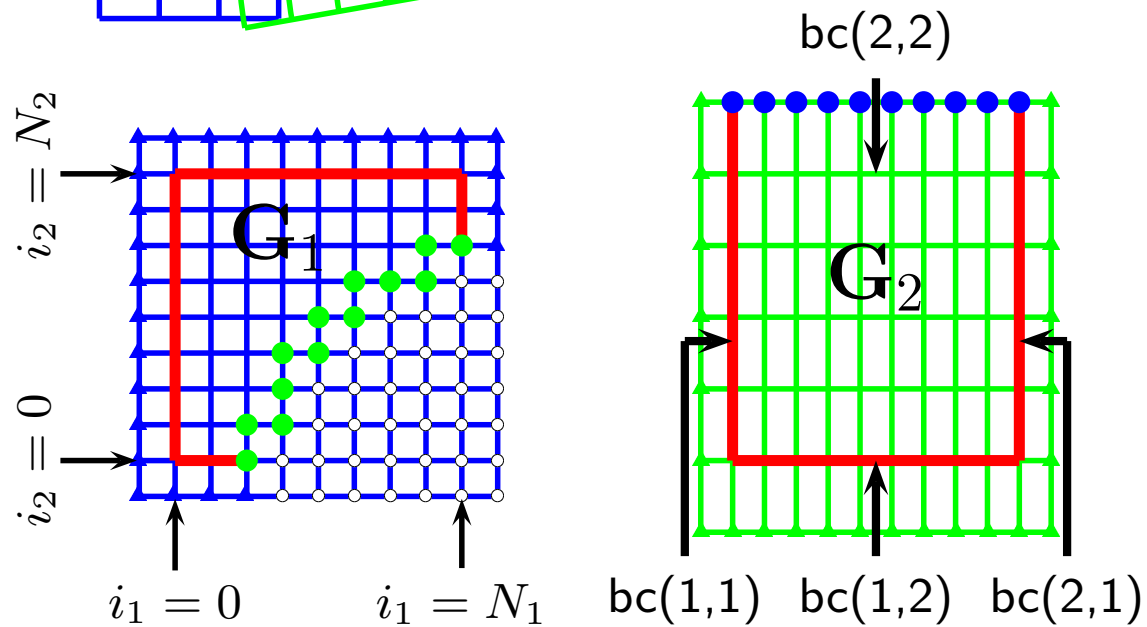
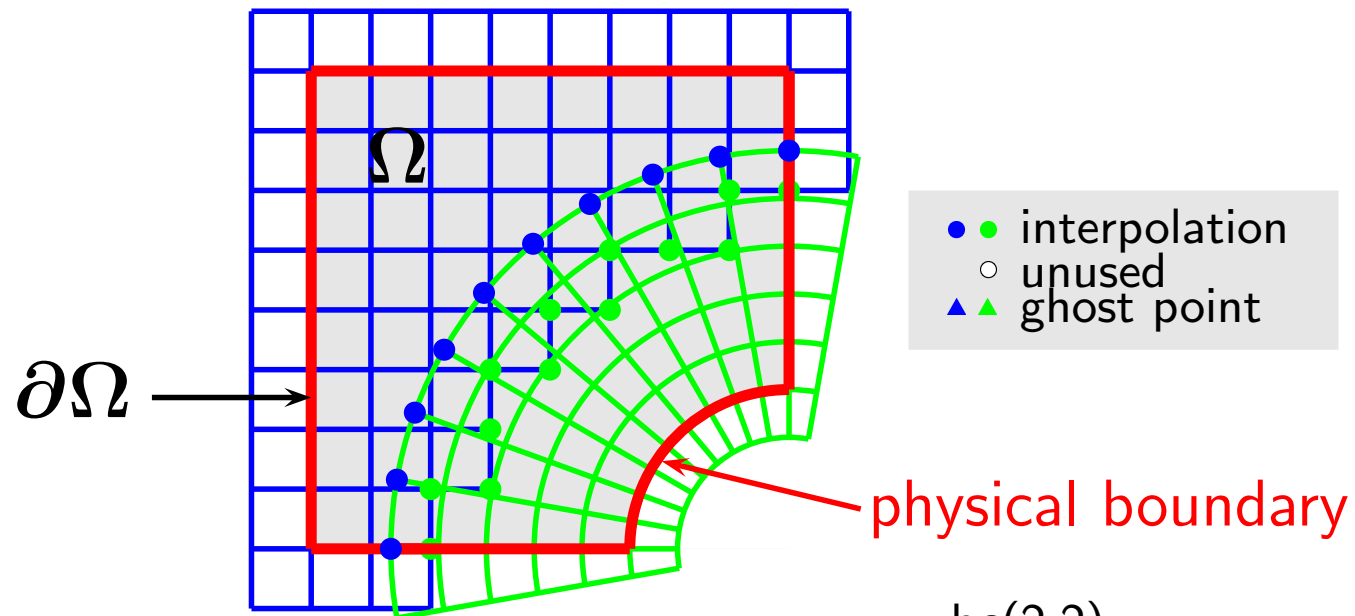




Sample hybrid grids



# Components of an Overlapping Grid



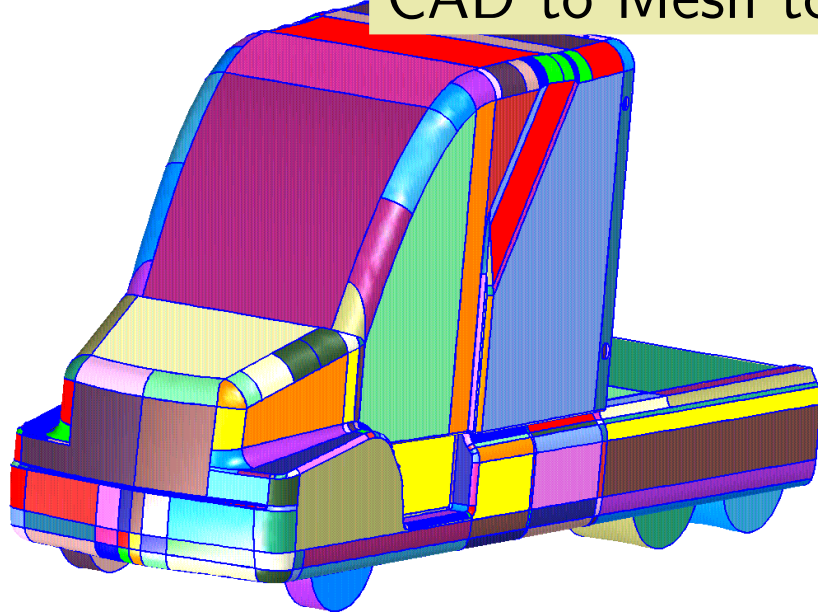
## Overture supports a high-level C++ interface (but is built mainly upon Fortran kernels):

Solve  $u_t + au_x + bu_y = \nu(u_{xx} + u_{yy})$

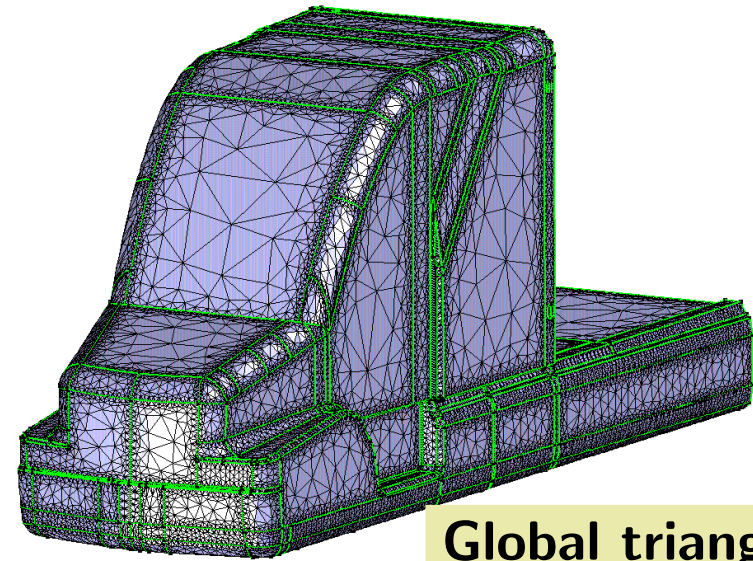
```
CompositeGrid cg; // create a composite grid
getFromADatabaseFile(cg,"myGrid.hdf");
floatCompositeGridFunction u(cg); // create a grid function
u=1.;
CompositeGridOperators op(cg); // operators
u.setOperators(op);
float t=0, dt=.005, a=1., b=1., nu=.1;
for( int step=0; step<100; step++ )
{
    u+=dt*( -a*u.x()-b*u.y()+nu*(u.xx()+u.yy()) ); // forward Euler
    t+=dt;
    u.interpolate();
    u.applyBoundaryCondition(0,dirichlet,allBoundaries,0.);
    u.finishBoundaryConditions();
}
```



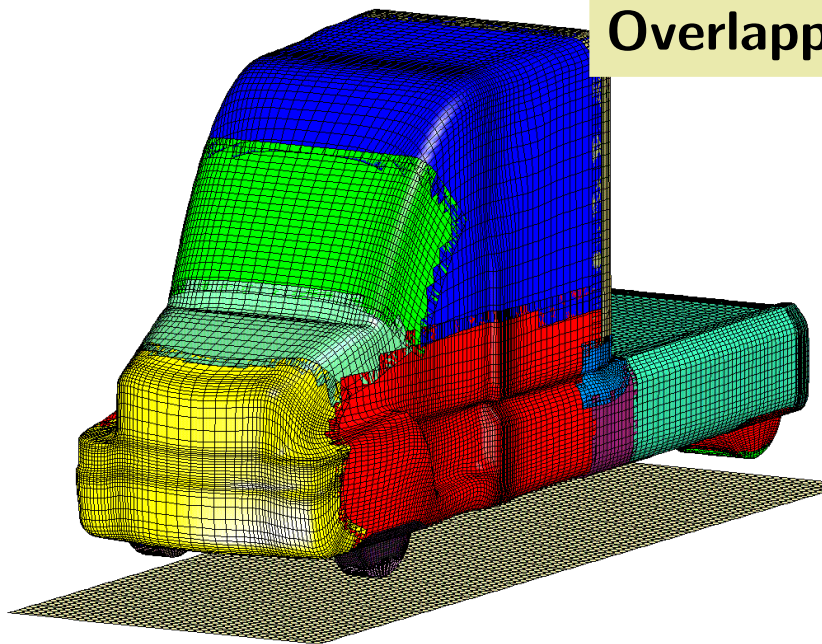
# CAD to Mesh to Solution with Overture



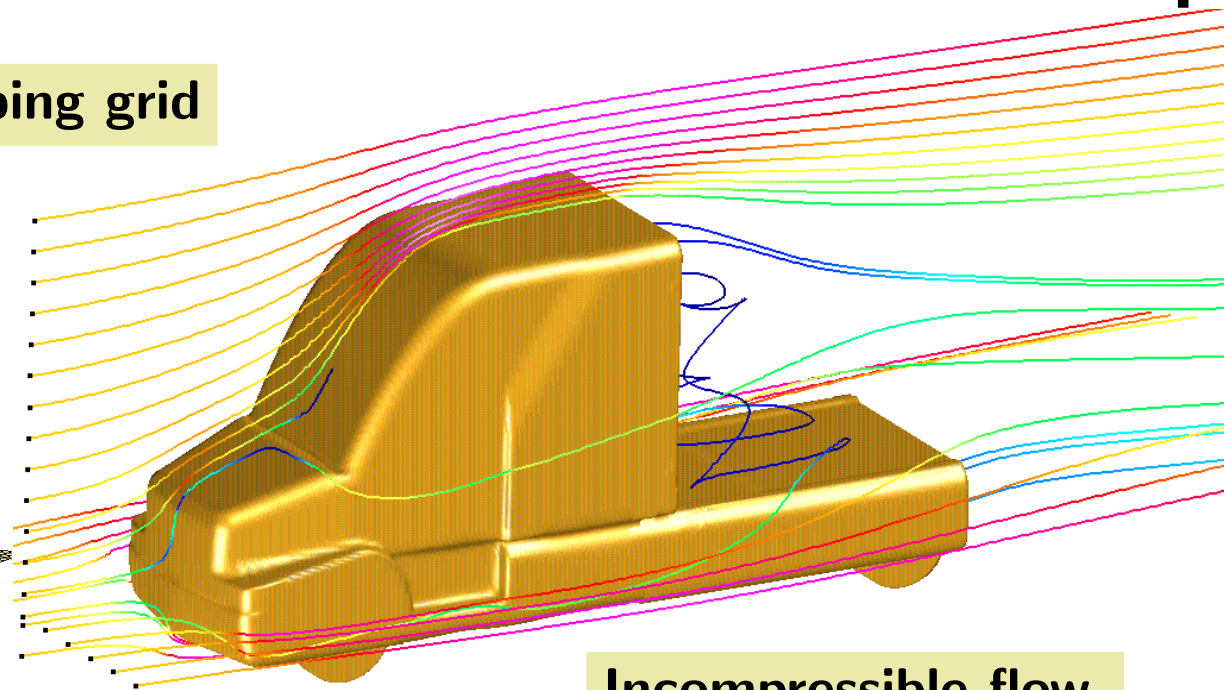
Cad fixup



Global triangulation



Overlapping grid



Incompressible flow.